1. **Document**: some text.
2. **Corpus**: a collection of documents.
3. **Vector**: a mathematically convenient representation of a document.
4. **Model**: an algorithm for transforming vectors from one representation to another.

**Corpus**

* A *corpus* is a collection of **Document** objects.
* Corpora serve two roles in Gensim:

1. **Input for training a Model.** 
   1. During training, the models use this training corpus to look for common themes and topics, initializing their internal model parameters.
   2. Gensim focuses on unsupervised models so that no human intervention, such as costly annotations or tagging documents by hand, is required.
2. **Documents to organize.**
   1. After training, a topic model can be used to extract topics from new documents (documents not seen in the training corpus).

* Such corpora can be indexed for **Similarity Queries**, queried by semantic similarity, clustered etc.

**Collect Documents into a Corpus**

documents = [

"Human machine interface for lab abc computer applications",

"A survey of user opinion of computer system response time",

"The EPS user interface management system",

"System and human system engineering testing of EPS",

"Relation of user perceived response time to error measurement",

"The generation of random binary unordered trees",

"The intersection graph of paths in trees",

"Graph minors IV Widths of trees and well quasi ordering",

"Graph minors A survey",

]

* A corpus is simply a collection of Documents objects.
* Documents are some form of text (strings in python)
* This is a tiny corpus of nine documents, each consisting of only a single sentence.

**Preprocessing:**

* Preprocessing documents will vary depending based on the problem at hand.
* In this example, we
  + Tokenize (by splitting on whitespaces)
  + Lowercasing each word
  + Remove punctuation
  + Remove stop words
  + Remove words that don’t occur more than once

**from** **pprint** **import** pprint *# pretty-printer*

**from** **collections** **import** defaultdict

*# remove common words and tokenize*

stoplist = set('for a of the and to in'.split())

texts = [

[word **for** word **in** document.lower().split() **if** word **not** **in** stoplist]

**for** document **in** documents

]

*# remove words that appear only once*

frequency = defaultdict(int)

**for** text **in** texts:

**for** token **in** text:

frequency[token] += 1

texts = [

[token **for** token **in** text **if** frequency[token] > 1]

**for** text **in** texts

]

pprint(texts)

Out:

[['human', 'interface', 'computer'],

['survey', 'user', 'computer', 'system', 'response', 'time'],

['eps', 'user', 'interface', 'system'],

['system', 'human', 'system', 'eps'],

['user', 'response', 'time'],

['trees'],

['graph', 'trees'],

['graph', 'minors', 'trees'],

['graph', 'minors', 'survey']]

* We now define a document as a list of ‘features’ and not by the surface string form.
* How you get to the ‘features’ document is up to you.
* We take these tokenized, processed documents and make them into a dictionary using a **bag-of-words** approach.
* Keep in mind, every domain is different and may call for different preprocessing/features (garbage in, garbage out).

**Dictionary**

* This module implements the concept of a Dictionary.
* The **dictionary** assigns a **unique integer id** to all **words** in each **document** appearing in the **corpus**.
* This representation of the dictionary is described as a **bag-of-words** approach.
  + We can make a dictionary with the genism.corpora.Dictionary import.
  + It takes a tokenized list of words in it’s constructor

**>>> from** **gensim.corpora** **import** Dictionary

>>>

**>>>** texts = [['human', 'interface', 'computer']]

**>>>** dct = Dictionary(texts) *# initialize a Dictionary*

* The Dictionary class sweeps across the documents (tokenized texts), collecting word counts and relevant statistics.
* The dictionary now holds all unique words in the corpus, which forms our **vocabulary**.
  + In our example:
    - We see there are 12 total distinct words in the processed corpus

**Vectors**

* Each document will be represented by twelve numbers (ie., by a 12-D vector).

To convert documents to vectors, we’ll use a document representation called bag-of-words. In this representation, each document is represented by one vector where each vector element represents a question-answer pair, in the style of:

Question: How many times does the word system appear in the document?

Answer: Once.

It is advantageous to represent the questions only by their (integer) ids. The mapping between the questions and ids is called a dictionary:

* To see the mapping between words and their ids: